Java Script

**# Browser Console**

* Can execute JavaScript command directly in browser, using console panel.
* How to access: right click and inspect /or/ menu > more tools > developer tool
* To write multiple lines: use ‘Shift + Enter’ to jump on next line
* To delete all stored data of variables: right click on ‘reload’ and select ‘empty cache and hard reload’

# Browser Snippets

* can write multiple line of JS code.
* Developer tools > Sources > Snippets
* ‘ctrl + F’ : to find and replace

# Basic functions

* typeof ( xyz ): this return the data type of ‘xyz’
* prompt(‘enter name’): popup that will ask of input
* alert(“Hello”): just to give a popup message
* str.length : return the length of string
* str.slice(start,end): slice out substring ; start is inclusive but end is exclusive ; counting with 0
* str.toUpperCase(): capitalize all characters
* str.toLowerCase():
* .
* Math.random(): generate a random number between 0 and 1 up to 16 decimal places
* Math.floor(): // floor value
* Var: variable

# Variables

* 2 variable can be interchange with this method…> [a,b] = [b,a] // destructuring assingment.

# Array

* Var arr = [“ab”,”bc”,”cd”,”xyz”];
* Arr.length: return length of array
* Arr.includes(xyz): return true if ‘xyz’ is present in array
* Arr.push(“hello”): add the value at the end of array;
* Arr.pop()

# Bubbling

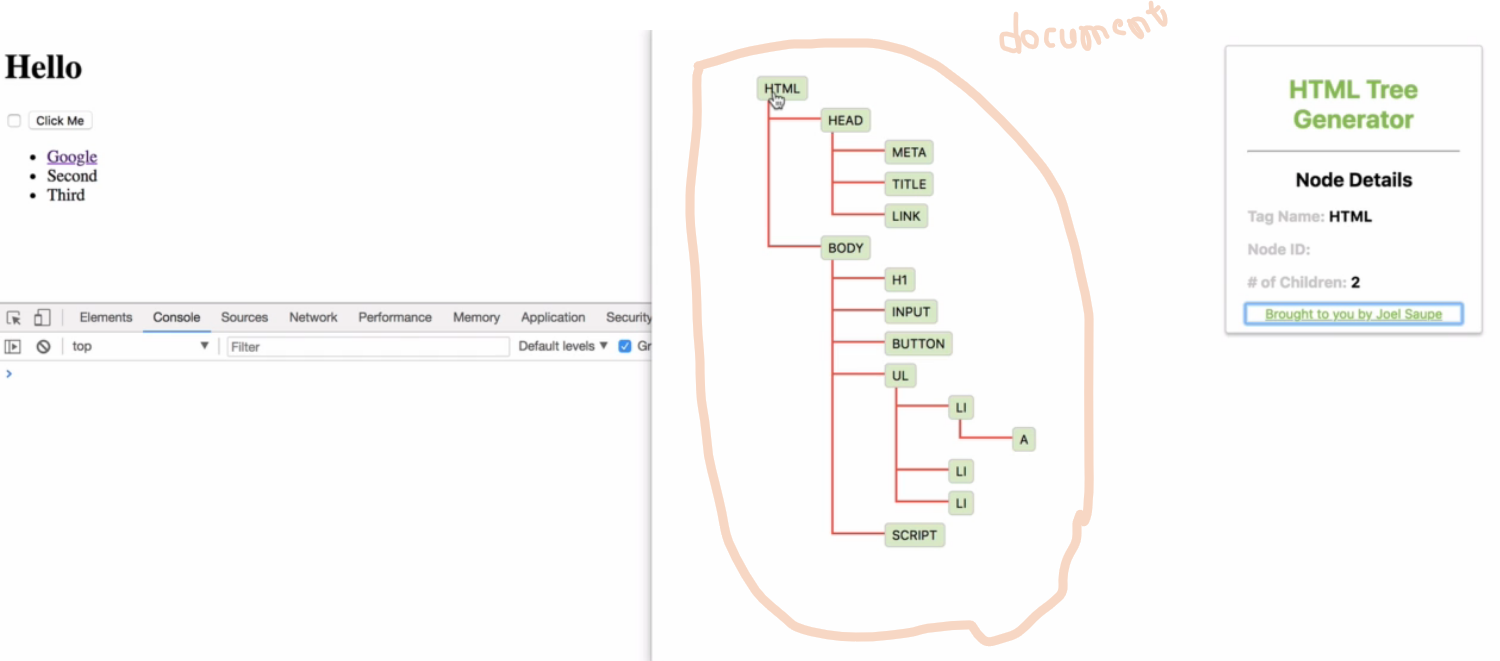
* Event.stopPropagation();

DOM

Document Object Model

It is helps in making html page dynamic.

It breaks entire html page into object and children form where root object is known as ‘document’ and we can access child and can perform action on them.



In above image Html is the first child of document

* DOM provides various method that can be use to access and manipulate various html element.
* Document is the root object, So we can use ‘.’ To access other child element
* Var Headtext = Document.firstChild.lastChild.firstElementChild; // will return heading tag and we can perform action on that.
* Headtext.innerHTML = “Anand”; // will change text in heading tag for above image.
* Headtext.style.color = “red”; // this will change the color of heading text.
* Document.querySelector(“h1”); // this will trigger first occurrence of element or “#id” or “.class”… we can add to make selection more precise…

(“li a”) // this will return a which is inside list tag. // consider space in between (hierarchical selector)

(“li.item”) // this will target “li” which has class “item”. // consider no space // both tag are of same element

* Document.querySelectorAll(“ ”); // similar to above but it will give array of all the matching element.
* Document.getElementsByTagName(“h1”); // this will return array of element with that name and that can be accessed sequentially as like array; // ‘Elements’ is plural in getelEmentsByTagName()
* getElementsByCLassName(); // same as the above one
* document.querySelector(“h1”).classList; // this will show all class attached to that element
* document.querySelector(“h1”).classList.add(“coder”); // this will add new class to that element
* document.querySelector(“h1”).classList.remove(“coder”); // this will remove particular class from element.
* document.querySelector(“h1”).classList.toggle(“coder”); // if exist then remove else add that class to eleme.
* document.querySelector(“a”).attributes; //

# Event Listeners

* Event Listener is a function that if a specific event happened on a page.
* addEventListener(input1 , input2): // inpu1 = what event it will trigger for; input2 = what will trigger
* ex: addEventListener(“click”,myFunc); // do not add angular bracket after function name.

# Higher Order Function

* Higher order function are function that can take other function as inputs.
* we can pass a function name to a variable and call that function through that variable…

# Object with constructor function

jQuery

We can download jQuery library or we can use CDN (content delivery network) to implement it.

Note: Make sure jQuery library get loaded before jQuery code start running… otherwise it will generate error.

* Either we can include jquery library and at the last of body tag sequentially or
* Use call back function library get ready first
* $(document).ready(function() { // … write your code here … });

To call jQuery methods we ither use ‘$()’ sign or ‘jQuery ()’

* $(‘h1’) // will triggered all h1 tags
* $(‘h1.cla) // will triggered all h1 tag which has class ‘cla’
* $(‘h1 atr’) //

jQuery have multiple function which are useful

Note: generally, if a function has single parameter, means it will return value related to that parameter.

If a function has two parameters, means it will set second parameter first attribute …

* $(‘h1’).css(‘color’); // this has single value so it will return value
* $(‘h1’).css(‘color’,’red’); // this have two parameter so it will set value
* $(‘h1’).addClass(‘h1style’);
* $(‘h1’).removeClass(‘h1style’);
* $(‘h1’).hasClass(‘h1style’); // will return true or false //as per class is applied or not
* $(‘h1’).text(“bye”); // change text as it is given
* $(‘h1’).html(“<em>bye</em>”); //this will also change text but we can also add html tags
* $(‘h1’).attr(“href”);
* $(‘h1’).attr(“href”,”google.com”);

Event Listeners

<https://www.minifier.org> // shorten the code to save space and memory

ANGULAR

* Npm install -g @angular/cli
* Ng new myapp
* Cd myapp
* Ng serve

// Question

Q1) capitalize first character of multiple words sentence

Q) JavaScript bobbling

Q) jQuery without CDN?

Q) jQuery Last 4 videos….

////// Deep Concepts

#

The reason why you don't need '()' in .length is because this is a **property** of an Object; whilst .toLowerCase() is a **method** of a String Object.  
  
Allow me to present an example:  
  
var text = "Hello World!";

Here we declare a variable **text** and assign the string "**Hello World!**" to it. This now becomes a [String Object.](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String)

If I want to change the string to all lowercase, then I can access the String Object's **method**(a function of the object) that could help me with this.

console.log(text.toLowerCase()); <- This would output "**hello world!**"

However, if I don't want to change the string, rather, I'd like to access some of its characteristics, then I'll tap into the length **property**.

console.log(text.length); <- This would output **12**

An easy way to remember this is to know that properties are certain characteristics that do not change the object, rather, they display some sort of information about it. Methods would perform a function, like changing all characters to lowercase.

///////////////////////////////////////////////////////////////////////////////////